

**MODUL 6**  
**ABSTRACTION**



**Adam Arthur Faizal**

**M3119001**

**TI A**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH VOKASI**

**UNIVERSITAS SEBELAS MARET**

**SURAKARTA**

**2020**

## **MODUL 6. Abstraksi**

### **Capaian Pembelajaran Praktikum:**

- Menerapkan konsep abstraksi dalam program
- Menerapkan konstruktor berparameter
- Menerapkan teknik casting
- Accessing method at another class

### **Tools:**

1. Java Development Kit (JDK)
2. Greenfoot IDE

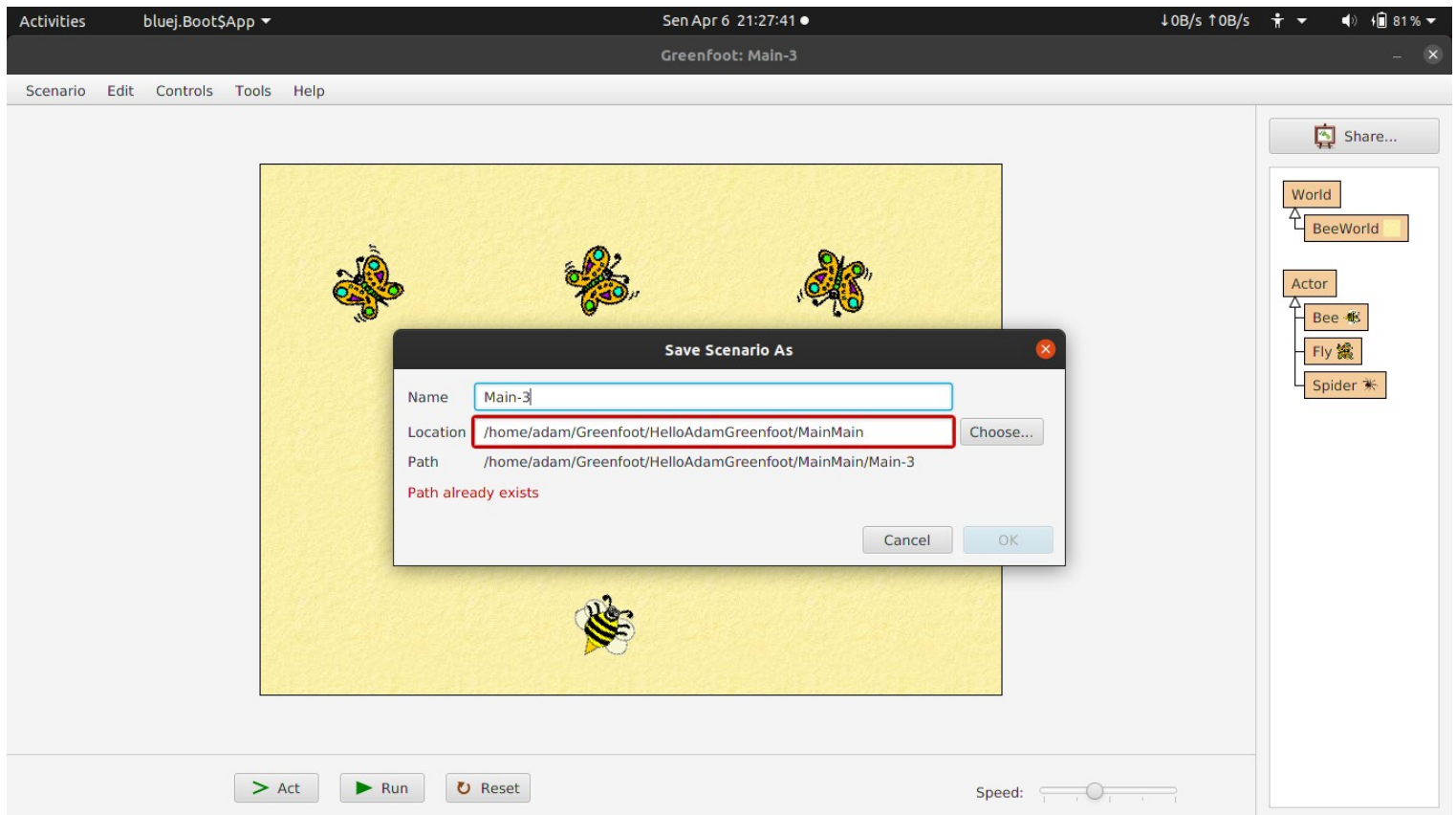
### **Terminologi:**

Isikan terminology yang sesuai untuk definisi dibawah ini:

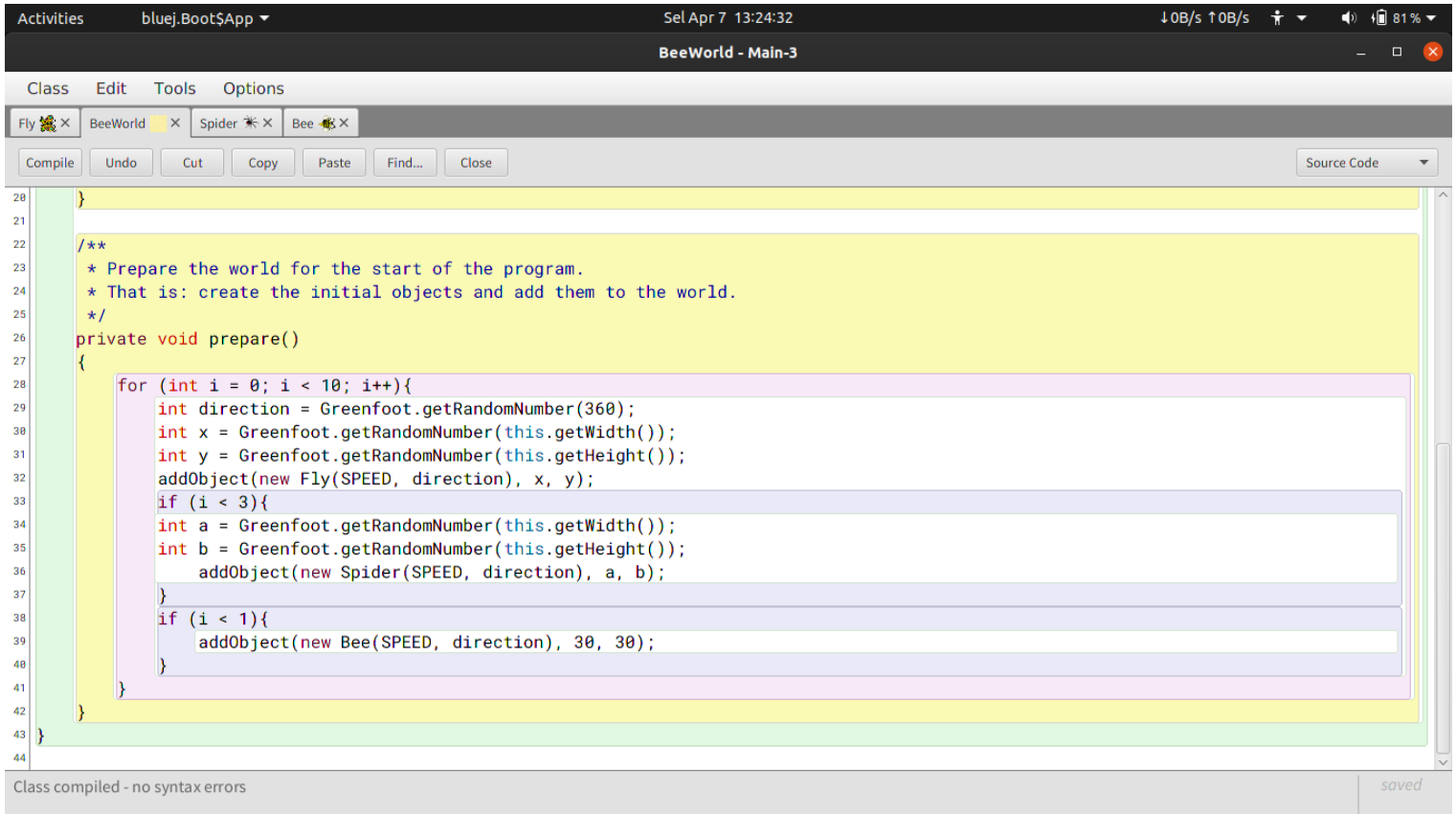
- |               |  |
|---------------|--|
| [Abstraction] | A technique used to command newly-created instances to perform different actions.                |
| [Casting]     | A technique used to tell java that a class is to be considered as another class                  |
| [Constructor] | A special method that is executed automatically whenever a new instance of the class is created. |

## TRY IT / SOLVE IT:

1. Buka scenario modul sebelumnya, kemudian lakukan versioning dengan melakukan save as. Beri nama folder yang baru untuk menyimpan versi scenario yang baru.



2. Dengan menggunakan abstraksi, modifikasi class `BeeWorld` dengan membuat method `prepare()` untuk menambahkan 10 objek `Fly()` ke dunia secara acak (X dan Y nya) dengan menggunakan perulangan, 1 objek `Bee()` pada lokasi 30,30, dan 3 objek `Spider()` secara acak dengan perulangan. Kemudian panggil method tersebut di konstruktor `BeeWorld`.

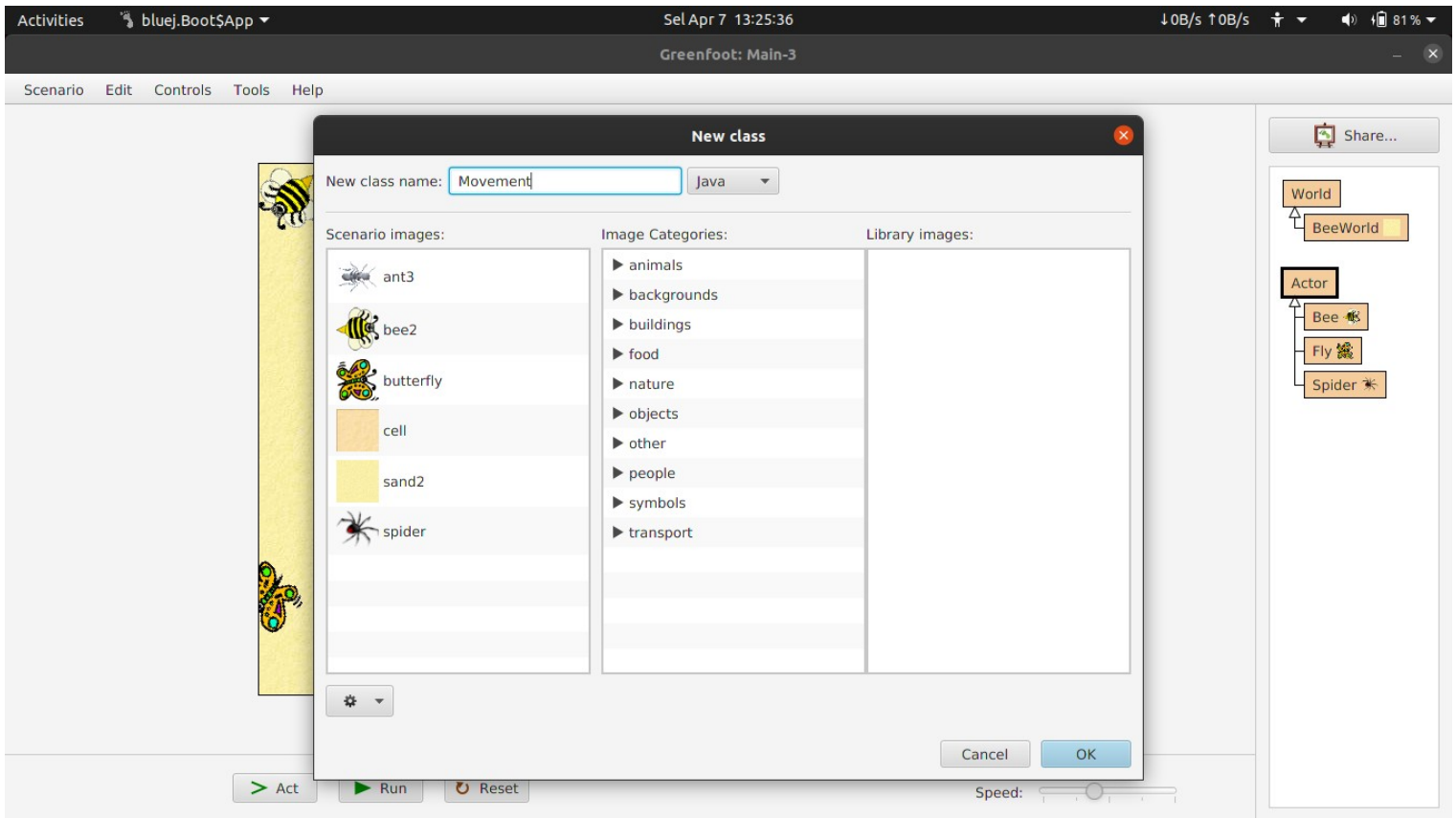


The screenshot shows a Java IDE window titled "BeeWorld - Main-3". The code editor displays the `BeeWorld` class with a new `prepare()` method. The method is annotated with a Javadoc comment: `/** * Prepare the world for the start of the program. * That is: create the initial objects and add them to the world. */`. The `prepare()` method is a private void method that uses a `for` loop to create 10 `Fly` objects. Inside the loop, there are two conditional blocks: one for `if (i < 3)` that creates 3 `Spider` objects, and another for `if (i < 1)` that creates 1 `Bee` object at coordinates (30, 30). The IDE interface includes a menu bar (Class, Edit, Tools, Options), a toolbar (Compile, Undo, Cut, Copy, Paste, Find..., Close), and a status bar at the bottom indicating "Class compiled - no syntax errors" and "saved".

```
20 }
21
22 /**
23  * Prepare the world for the start of the program.
24  * That is: create the initial objects and add them to the world.
25  */
26 private void prepare()
27 {
28     for (int i = 0; i < 10; i++){
29         int direction = Greenfoot.getRandomNumber(360);
30         int x = Greenfoot.getRandomNumber(this.getWidth());
31         int y = Greenfoot.getRandomNumber(this.getHeight());
32         addObject(new Fly(SPEED, direction), x, y);
33         if (i < 3){
34             int a = Greenfoot.getRandomNumber(this.getWidth());
35             int b = Greenfoot.getRandomNumber(this.getHeight());
36             addObject(new Spider(SPEED, direction), a, b);
37         }
38         if (i < 1){
39             addObject(new Bee(SPEED, direction), 30, 30);
40         }
41     }
42 }
43
44 }
```

Class compiled - no syntax errors saved

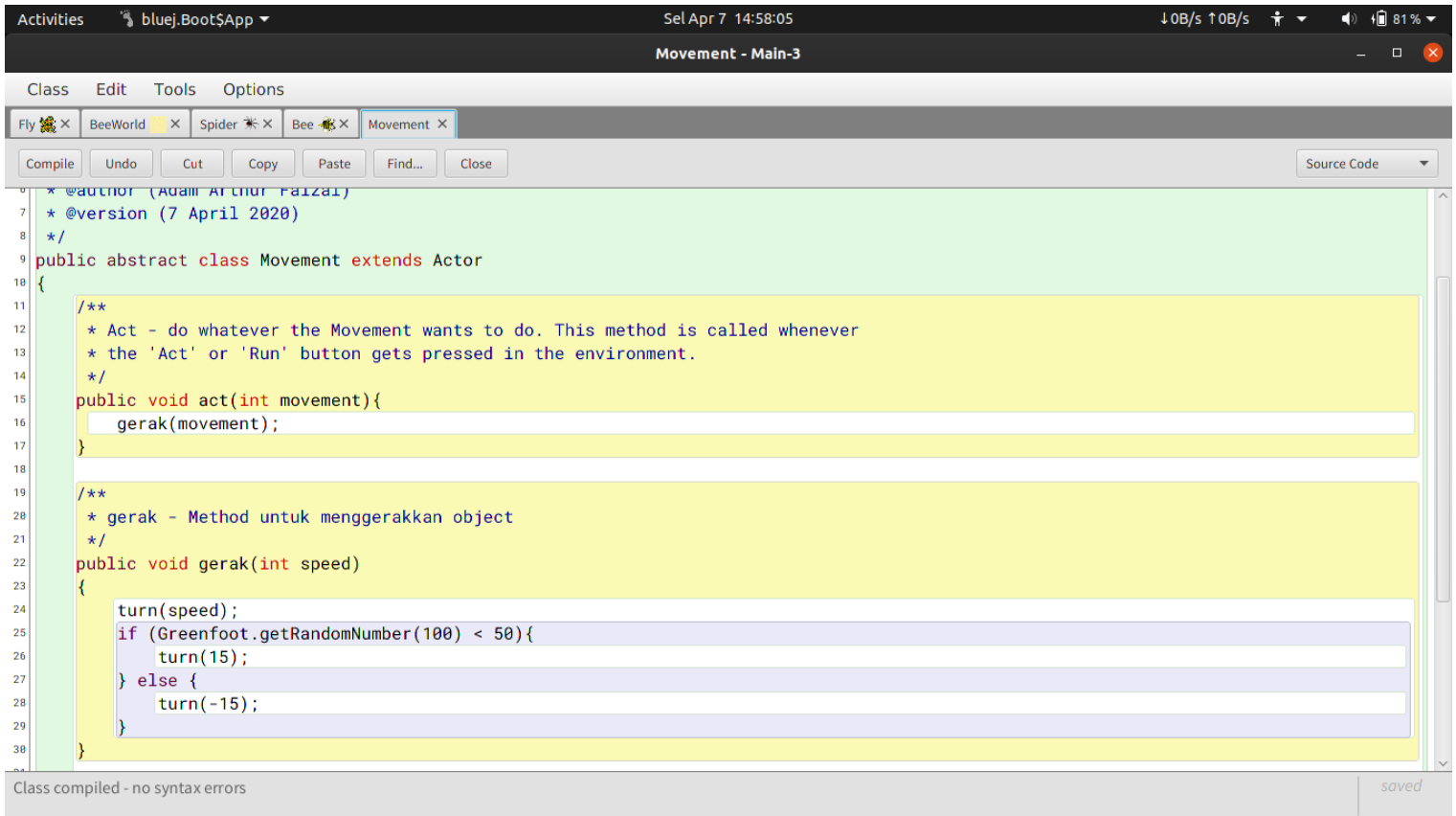
### 3. Buat class Movement sebagai sub class Actor. (tanpa image)



### 4. Modifikasi kelas Spider dan Fly, yang awalnya mengindik (extends) ke kelas Aktor, anda pindahkan mengindik pada kelas Movement.

```
9 public class Fly extends Movement
9 public class Bee extends Movement
public class Spider extends Movement
```

5. Buatlah method gerak() di kelas Movement yang akan mengganti gerak Spider dan Fly, dimana pergerakan dari objek Spider dan Fly adalah bergerak maju dengan kecepatan masing-masing (speed) dan berbelok ke kanan dan ke kiri sebesar 10 secara random dengan probabilitas 50:50.

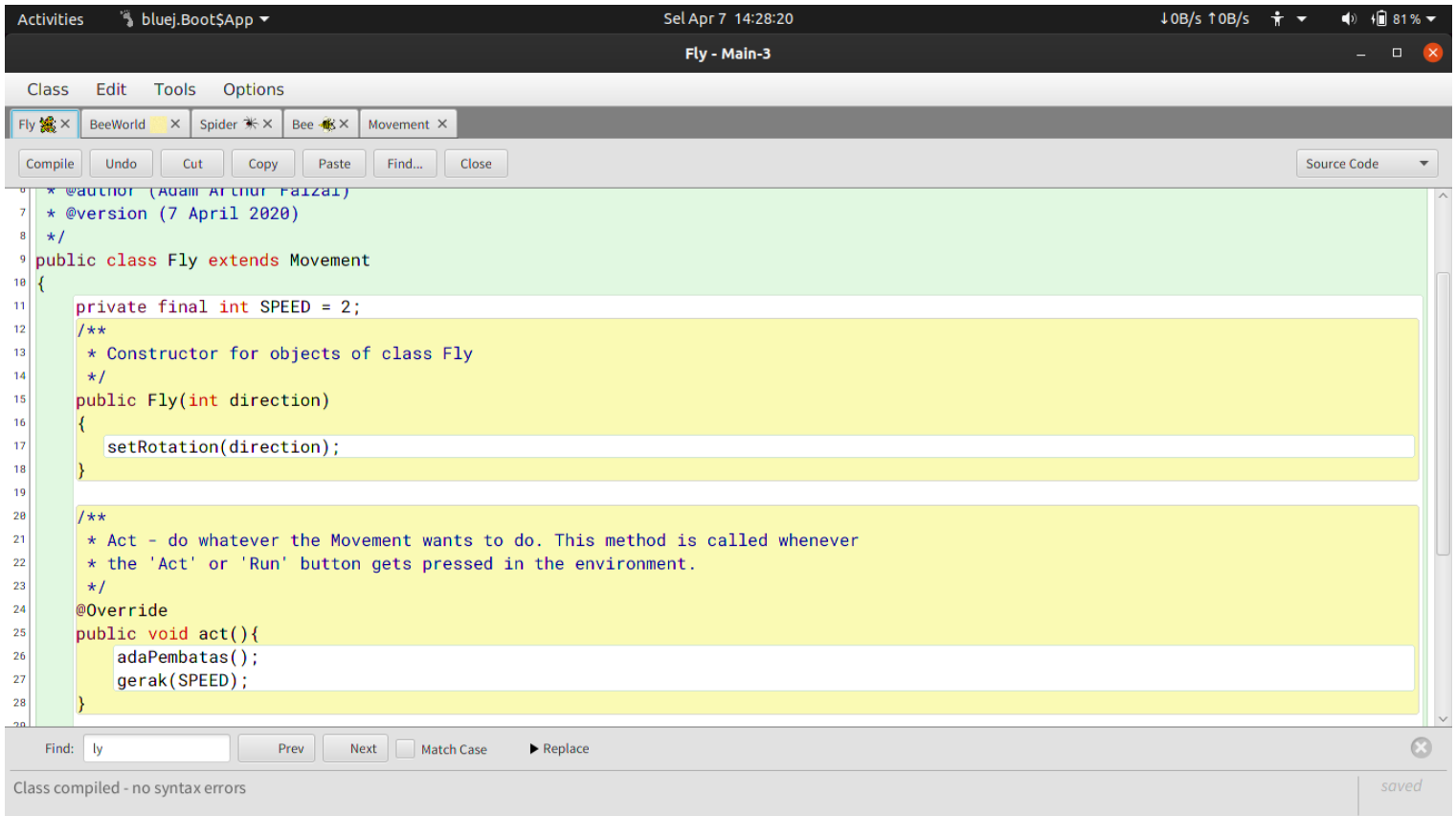


The screenshot shows a Java IDE window titled "Movement - Main-3". The code editor displays the following Java code for the Movement class:

```
1  * @author (Adam Arthur Faizal)
2  * @version (7 April 2020)
3  */
4  public abstract class Movement extends Actor
5  {
6
7      /**
8       * Act - do whatever the Movement wants to do. This method is called whenever
9       * the 'Act' or 'Run' button gets pressed in the environment.
10     */
11     public void act(int movement){
12         gerak(movement);
13     }
14
15     /**
16     * gerak - Method untuk menggerakkan object
17     */
18     public void gerak(int speed)
19     {
20         turn(speed);
21         if (Greenfoot.getRandomNumber(100) < 50){
22             turn(15);
23         } else {
24             turn(-15);
25         }
26     }
27 }
```

At the bottom of the IDE, a status bar indicates "Class compiled - no syntax errors" and a "saved" button is visible on the right.

6. **Panggil / invoke** method gerak() di method act Spider maupun Fly. (Speed untuk Spider adalah 3 dan untuk Fly sebesar 2) - **Gunakan variabel**

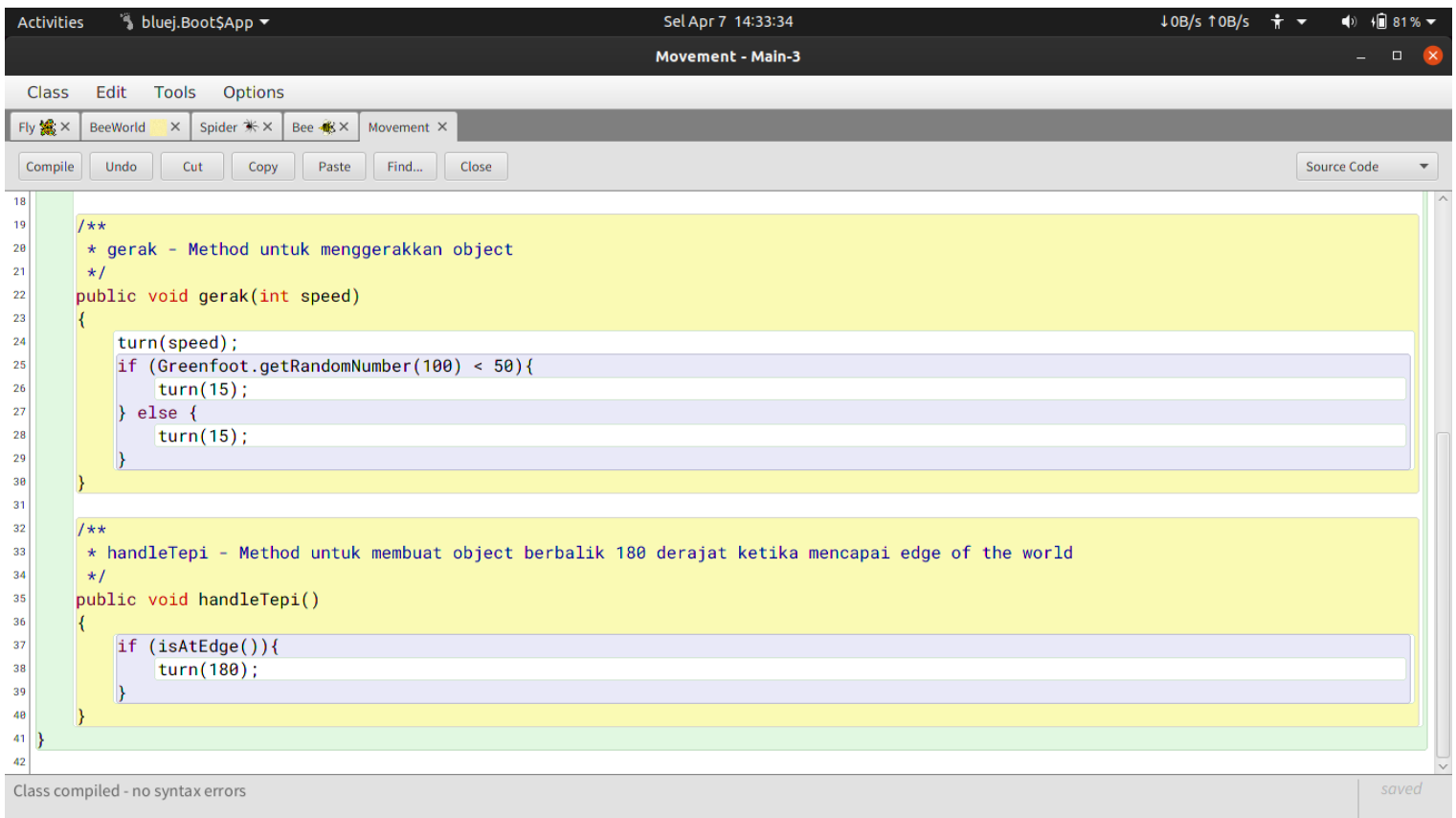


The screenshot shows a Java IDE window titled "Fly - Main-3". The code editor displays the following Java code for the Fly class:

```
6  * @author (Adam Arthur Faizal)
7  * @version (7 April 2020)
8  */
9  public class Fly extends Movement
10 {
11     private final int SPEED = 2;
12     /**
13      * Constructor for objects of class Fly
14      */
15     public Fly(int direction)
16     {
17         setRotation(direction);
18     }
19
20     /**
21      * Act - do whatever the Movement wants to do. This method is called whenever
22      * the 'Act' or 'Run' button gets pressed in the environment.
23      */
24     @Override
25     public void act(){
26         adaPembatas();
27         gerak(SPEED);
28     }
29 }
```

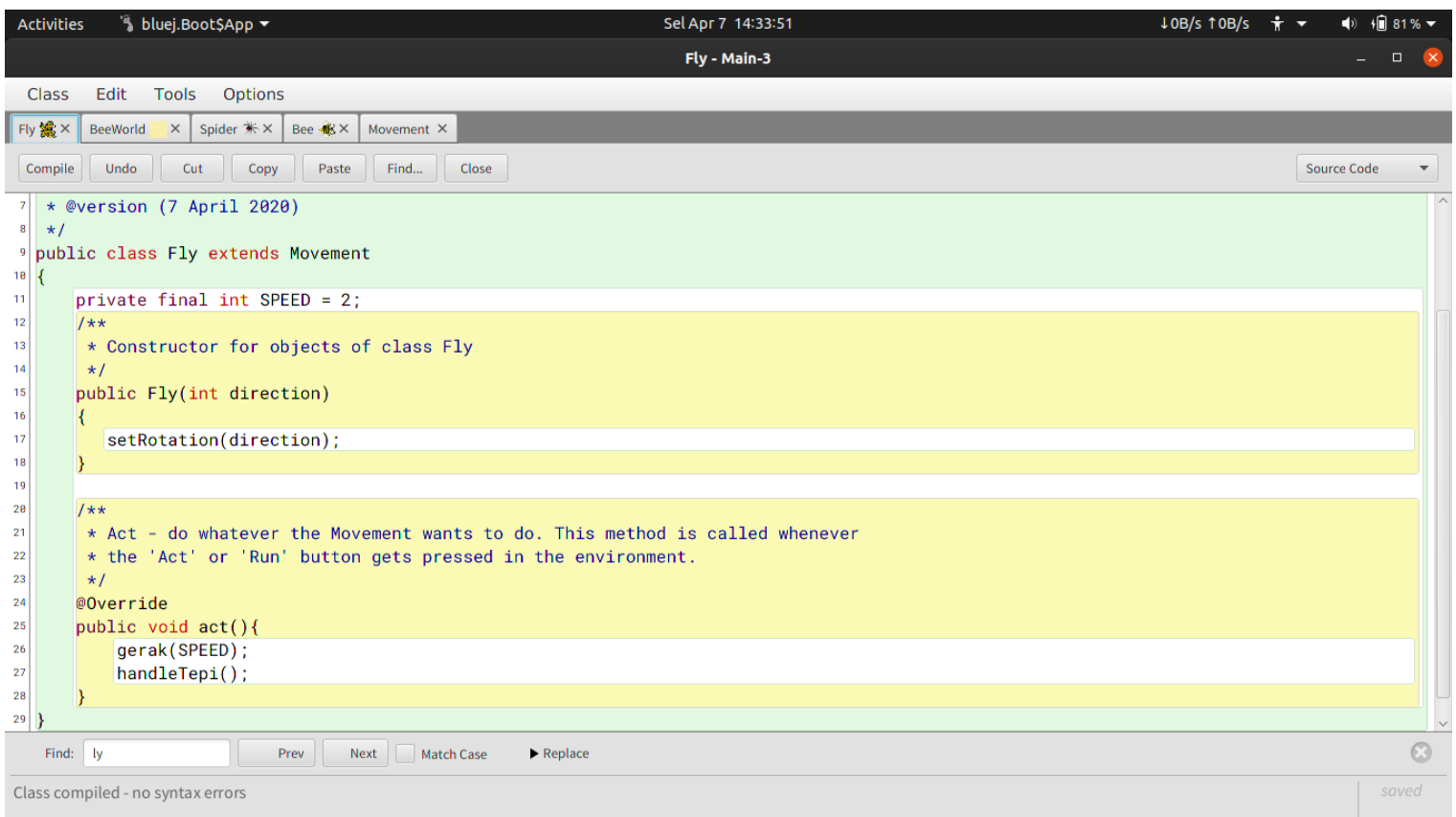
At the bottom of the IDE, there is a search bar with the text "ly" and buttons for "Prev", "Next", "Match Case", and "Replace". Below the search bar, it says "Class compiled - no syntax errors" and "saved".

7. Modifikasi program anda, **Tarik/pindahkan** method *handleTepi()* pada class Spider ke class Movement. Kemudian panggil method *handleTepi()* di method *act()* pada kelas Fly dan kelas Spider.



The screenshot shows the BlueJ IDE with the 'Movement' class selected. The source code is displayed in the main editor area. The code defines two methods: *gerak()* and *handleTepi()*. The *gerak()* method takes an integer *speed* and calls *turn(speed)*. The *handleTepi()* method calls *isAtEdge()* and *turn(180)* if the object is at the edge. The status bar at the bottom indicates 'Class compiled - no syntax errors' and 'saved'.

```
18
19 /**
20  * gerak - Method untuk menggerakkan object
21  */
22 public void gerak(int speed)
23 {
24     turn(speed);
25     if (Greenfoot.getRandomNumber(100) < 50){
26         turn(15);
27     } else {
28         turn(15);
29     }
30 }
31
32 /**
33  * handleTepi - Method untuk membuat object berbalik 180 derajat ketika mencapai edge of the world
34  */
35 public void handleTepi()
36 {
37     if (isAtEdge()){
38         turn(180);
39     }
40 }
41
42 }
```



The screenshot shows the BlueJ IDE with the 'Fly' class selected. The source code is displayed in the main editor area. The code defines the *Fly* class, which extends *Movement*. It includes a constant *SPEED*, a constructor *Fly(int direction)* that calls *setRotation(direction)*, and an *act()* method that calls *gerak(SPEED)* and *handleTepi()*. The status bar at the bottom indicates 'Class compiled - no syntax errors' and 'saved'.

```
7  * @version (7 April 2020)
8  */
9 public class Fly extends Movement
10 {
11     private final int SPEED = 2;
12     /**
13      * Constructor for objects of class Fly
14      */
15     public Fly(int direction)
16     {
17         setRotation(direction);
18     }
19
20     /**
21      * Act - do whatever the Movement wants to do. This method is called whenever
22      * the 'Act' or 'Run' button gets pressed in the environment.
23      */
24     @Override
25     public void act(){
26         gerak(SPEED);
27         handleTepi();
28     }
29 }
```



Activities bluej.Boot\$App Sel Apr 7 14:35:13 ↓0B/s ↑0B/s 81%

Spider - Main-3

Class Edit Tools Options

Fly × BeeWorld × Spider × Bee × Movement ×

Compile Undo Cut Copy Paste Find... Close Source Code

```
6  * @author (Adam Arthur Faizal)
7  * @version (7 April 2020)
8  */
9  public class Spider extends Movement
10 {
11     private final int SPEED = 3;
12     /**
13      * Constructor for objects of class Spider
14      */
15     public Spider(int direction)
16     {
17         setRotation(direction);
18     }
19
20     /**
21      * Act - do whatever the Movement wants to do. This method is called whenever
22      * the 'Act' or 'Run' button gets pressed in the environment.
23      */
24     @Override
25     public void act(){
26         gerak(SPEED);
27         handleTepi();
28     }
29 }
30
```

Class compiled - no syntax errors saved

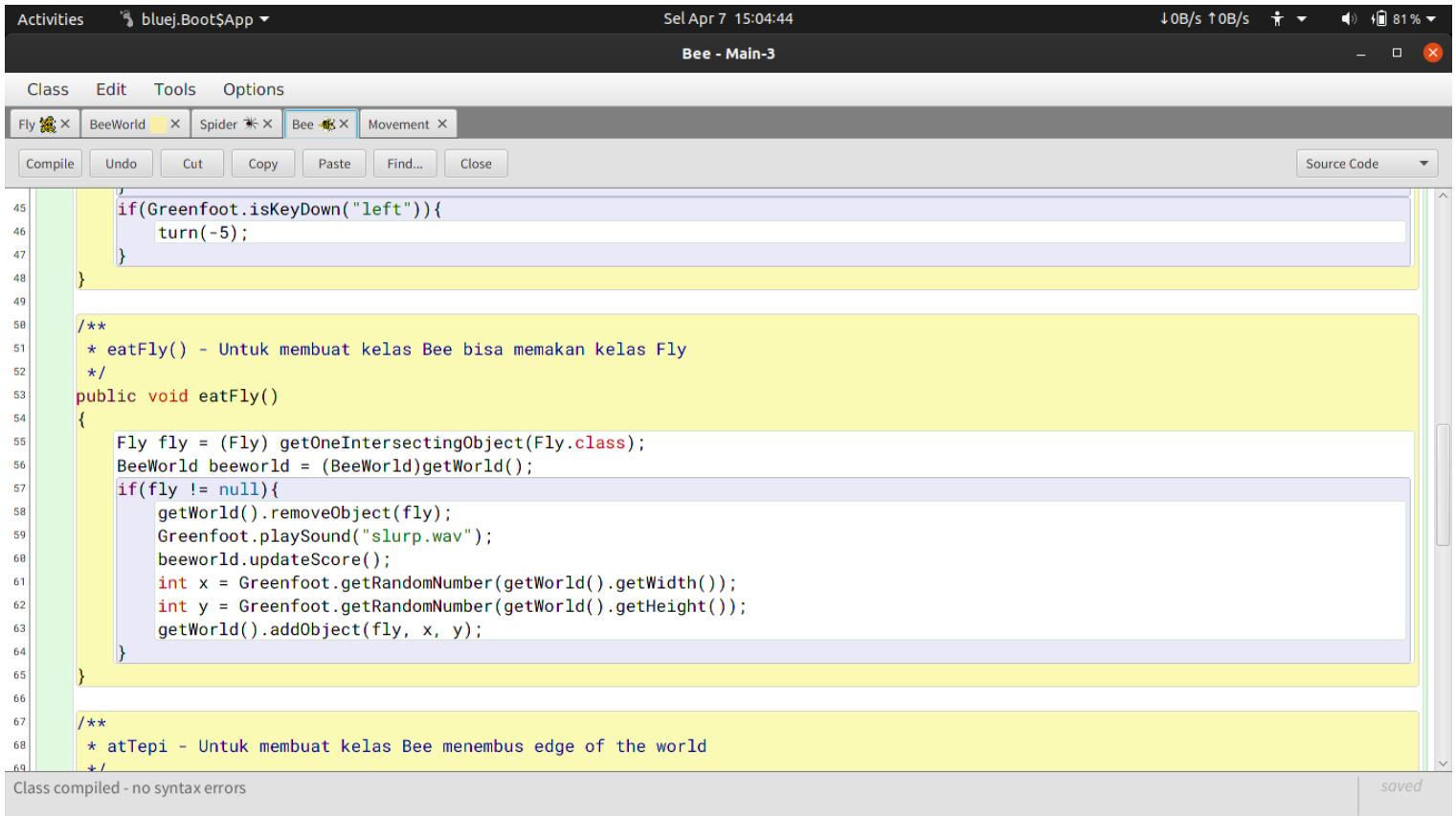
8. Tambahkan variabel score pada kelas BeeWorld. Beri nilai awal score dengan Nol. Kemudian buatlah method *updateScore()* di kelas BeeWorld yang berfungsi untuk menambahkan score ketika Bee memakan objek Fly serta menampilkan nilai Score pada layar.

```
29 int x = Greenfoot.getRandomNumber(this.getWidth());
30 int y = Greenfoot.getRandomNumber(this.getHeight());
31 addObject(new Fly(direction), x, y);
32 if (i < 3){
33     int a = Greenfoot.getRandomNumber(this.getWidth());
34     int b = Greenfoot.getRandomNumber(this.getHeight());
35     addObject(new Spider(direction), a, b);
36 }
37 if (i < 1){
38     addObject(new Bee(direction), 30, 30);
39 }
40 }
41 }
42 /**
43  * updateScore - Method untuk memperbarui nilai score ketika kelas Bee memakan kelas Fly
44  */
45 public void updateScore()
46 {
47     scores++;
48     showText("Score : " + scores, 60, 480);
49 }
50 }
51 }
52 }
53 }
```

Error(s) found in class.  
Press Ctrl+K or click link on right to go to next error.

saved

9. Panggil method `updateScore()` di kelas Bee pada lokasi yang sesuai untuk menambahkan score ketika Bee memakan Fly. (Gunakan Teknik casting)



```
45     if(Greenfoot.isKeyDown("left")){
46         turn(-5);
47     }
48 }
49
50 /**
51  * eatFly() - Untuk membuat kelas Bee bisa memakan kelas Fly
52  */
53 public void eatFly()
54 {
55     Fly fly = (Fly) getOneIntersectingObject(Fly.class);
56     BeeWorld beeworld = (BeeWorld)getWorld();
57     if(fly != null){
58         getWorld().removeObject(fly);
59         Greenfoot.playSound("slurp.wav");
60         beeworld.updateScore();
61         int x = Greenfoot.getRandomNumber(getWorld().getWidth());
62         int y = Greenfoot.getRandomNumber(getWorld().getHeight());
63         getWorld().addObject(fly, x, y);
64     }
65 }
66
67 /**
68  * atTepi - Untuk membuat kelas Bee menembus edge of the world
69  */

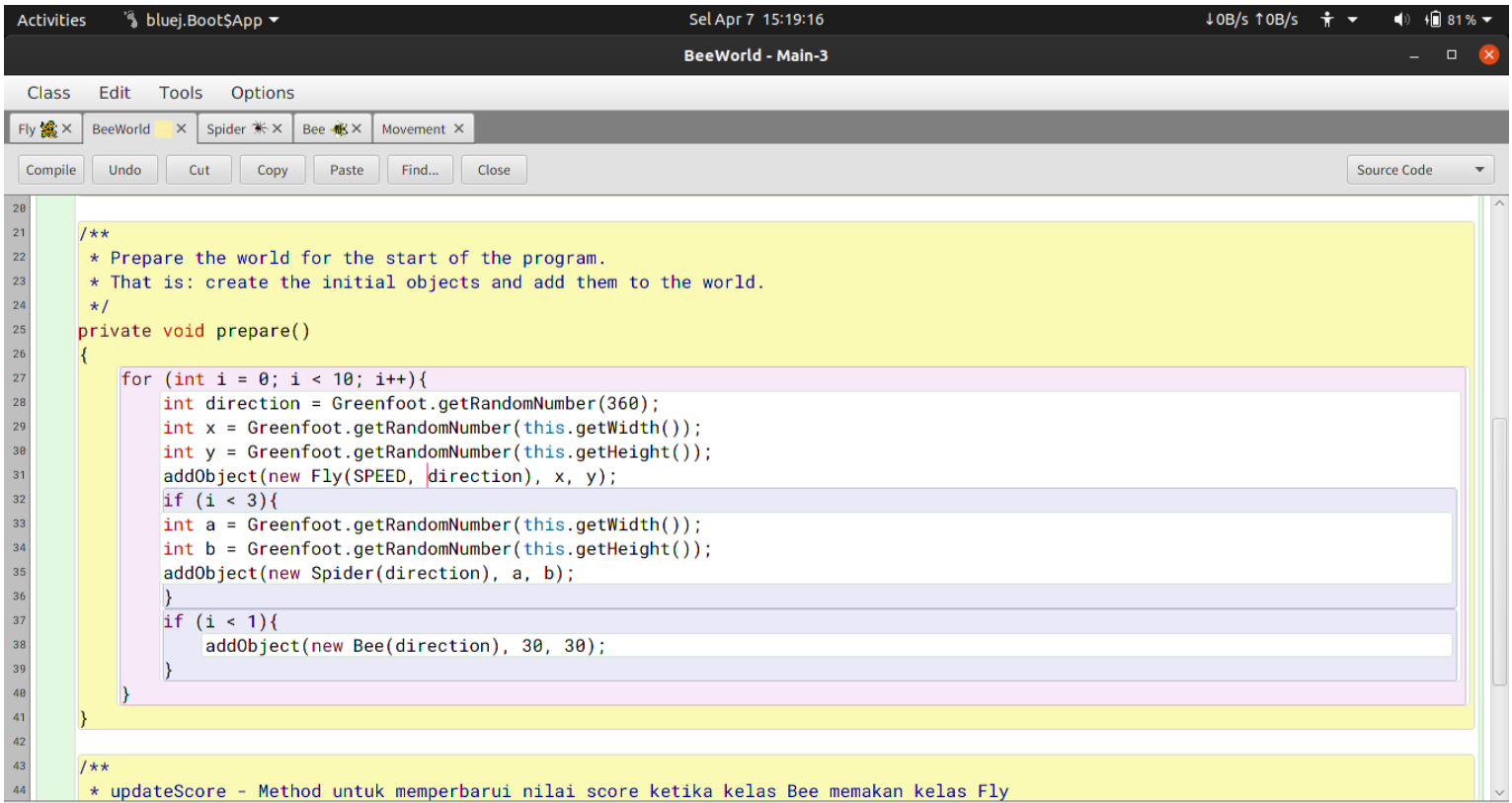
```

Class compiled - no syntax errors

saved

Tantangan:

10. Setiap objek Fly yang diciptakan akan memiliki kecepatannya masing-masing (minimal 1 dan max 4). Modifikasi program anda sehingga requirement ini bisa terpenuhi. (Gunakan konsep konstruktor berparameter)



```
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44

/**
 * Prepare the world for the start of the program.
 * That is: create the initial objects and add them to the world.
 */
private void prepare()
{
    for (int i = 0; i < 10; i++){
        int direction = Greenfoot.getRandomNumber(360);
        int x = Greenfoot.getRandomNumber(this.getWidth());
        int y = Greenfoot.getRandomNumber(this.getHeight());
        addObject(new Fly(SPEED, direction), x, y);
        if (i < 3){
            int a = Greenfoot.getRandomNumber(this.getWidth());
            int b = Greenfoot.getRandomNumber(this.getHeight());
            addObject(new Spider(direction), a, b);
        }
        if (i < 1){
            addObject(new Bee(direction), 30, 30);
        }
    }
}

/**
 * updateScore - Method untuk memperbarui nilai score ketika kelas Bee memakan kelas Fly
```

**Setelah sesi praktikum SELESAI, laporan praktikum dan source code (zip) harus dikirim/diupload ke google classroom sebelum pertemuan berikutnya.**